文章编号: 1000-0887(2008)06-0726-09

凸二次整数规划的随机水平值逼近算法*

彭 拯[1,2], 邬冬华[1]

(1. 上海大学 数学系, 上海 200444;
2. 湖南理工学院 数学系, 湖南岳阳 414006)

(刘曾荣推荐)

摘要: 对凸二次整数极小化问题提出了一种随机水平值逼近算法, 该算法应用了重点取样技术, 并利用极小化相对熵的思想来更新取样密度. 对算法的渐近收敛性进行了证明, 给出了数值实验的结果.

关 键 词: 凸二次整数极小化; 随机水平值逼近; 相对熵方法; 渐近收敛性
中图分类号: O221.2 文献标识码: A

引 言

整数规划在交通运输、生产计划、通讯设计和材料科学等诸方面有着广泛应用[1], 著名的"中国邮路问题"、背包问题等均被归结为整数规划问题; 此外, 整数规划问题在复杂性理论研究方面具有独特的重要价值, 并且极具挑战性, 因此它受到来自各个领域的研究者的普遍关注, 其中包括数学家、运筹学和经济学家、计算机专家乃至物理学家和化学家. 他们根据自身研究领域的特点创造了大量的算法来求解这类问题. 这些算法大体上可分为两类: 确定性算法和随机性算法. 确定性算法主要包括分枝定界法[2-3]与割平面法[4-5]及它们的各种改进与变形[6]. 随机性算法主要是概率启发式算法, 目前较为成熟有效的随机性算法为遗传算法、模拟退火算法与禁忌搜索算法等[7-10].

本文考虑凸二次整数极小化问题

$$(\mathrm{PF})\begin{cases}\min f_{\mathrm{P}}(\boldsymbol{x}) = \dfrac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{x} + \boldsymbol{d}^{\mathrm{T}}\boldsymbol{x}\\ \mathrm{s.t.}\ \boldsymbol{A}\boldsymbol{x} \leqslant \boldsymbol{b},\\ \boldsymbol{x} \in Z^n,\end{cases} \tag{1}$$

其中 $\boldsymbol{H} \in R^{n\times n}$ 是正半定矩阵, $\boldsymbol{A} \in R^{m\times n}$, $\boldsymbol{d} \in R^n$ 及 $\boldsymbol{b} \in R^m$.

我们针对问题(PF)提出一种随机水平值逼近算法(ISLVAM), 算法基于下面弱对偶定理.

问题(PF)的二次规划松弛为

* 收稿日期: 2007-07-09; 修订日期: 2008-04-30
基金项目: 国家自然科学基金资助项目(10671117); 上海市重点学科资助项目(J50101); 湖南省教育厅青年基金资助项目(06B037)
作者简介: 邬冬华(1961—)男, 上海人, 教授, 博士(联系人. Tel: +86-21-66132413; E-mail: dhwu@staff.shu.edu.cn).

$$(\mathrm{SF})\begin{cases}\min f_s(\boldsymbol{x}) = \dfrac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{H}\boldsymbol{x} + \boldsymbol{d}^{\mathrm{T}}\boldsymbol{x}\\ \text{s.t. } \boldsymbol{A}\boldsymbol{x} \leqslant \boldsymbol{b}.\end{cases} \tag{2}$$

二次规划(SF) Lagrange 对偶规划为

$$(\mathrm{DF})\begin{cases}\max f_d(\boldsymbol{u}) = -\dfrac{1}{2}\boldsymbol{u}^{\mathrm{T}}\boldsymbol{D}\boldsymbol{u} + \boldsymbol{u}^{\mathrm{T}}\boldsymbol{e} - \dfrac{1}{2}\boldsymbol{d}^{\mathrm{T}}\boldsymbol{H}^{-1}\boldsymbol{d}\\ \text{s.t. } \boldsymbol{u} \geqslant 0.\end{cases} \tag{3}$$

其中 $\boldsymbol{D} = -\boldsymbol{A}\boldsymbol{H}^{-1}\boldsymbol{A}^{\mathrm{T}}$, $\boldsymbol{e} = -\boldsymbol{b} - \boldsymbol{A}\boldsymbol{H}^{-1}\boldsymbol{d}$. 分别记问题(PF)、(SF)和(DF)的可行域为 S_{pf}、S_{sf}、S_{df}, 即 $S_{\mathrm{sf}} = \{\boldsymbol{x}: \boldsymbol{A}\boldsymbol{x} \leqslant \boldsymbol{b}\}$, $S_{\mathrm{pf}} = S_{\mathrm{sf}} \cap Z^n$, $S_{\mathrm{df}} = \{\boldsymbol{u}: \boldsymbol{u} \geqslant 0\}$. 由弱对偶定理[11]我们有

定理 1[11] 对 $\forall \boldsymbol{x} \in S_{\mathrm{sf}}$ 及 $\forall \boldsymbol{u} \in S_{\mathrm{df}}$, 不等式 $f_d(\boldsymbol{u}) \leqslant f_s(\boldsymbol{x})$ 成立.

定理 2[11] 若 $\exists \boldsymbol{x}^* \in S_{\mathrm{pf}}$, $\boldsymbol{u}^* \in S_{\mathrm{df}}$, 满足 $f_p(\boldsymbol{x}^*) = f_d(\boldsymbol{u}^*) = c^*$, 则 $\boldsymbol{x}^*$ 和 $\boldsymbol{u}^*$ 分别是问题(PF)和(DF)的最优解, c^* 是最优值.

若令 $\varepsilon(\boldsymbol{x}, \boldsymbol{u}) = f_p(\boldsymbol{x}) - f_d(\boldsymbol{u})$, $\boldsymbol{x} \in S_{\mathrm{pf}}$, $\boldsymbol{u} \in S_{\mathrm{df}}$, 那么由上述定理可知, 问题

$$\begin{cases}\min \varepsilon(\boldsymbol{x}, \boldsymbol{u})\\ \text{s.t. } \boldsymbol{x} \in Z^n \cap S_{\mathrm{sf}},\\ \boldsymbol{u} \in S_{\mathrm{df}}\end{cases} \tag{4}$$

的解 $(\boldsymbol{x}^*, \boldsymbol{u}^*)$ 中 $\boldsymbol{x}^*$ 就是 PF 的解.

对于连续优化问题, 我们提出了一种随机水平值逼近算法(SLVAM), 并研究了它的收敛性与计算复杂性. 数值实验表明, 算法(SLVAM)对于 Lipschitz 连续优化问题有较高的数值精度和较好的运算效率. 本文的目的是构造求解凸二次整数极小化问题的随机水平值逼近算法(ISLVAM), 证明该算法的收敛性, 并且通过数值实验说明算法的有效性. 这也是相对熵方法的文献[12]中所提出的一个研究方向.

本文按如下形式来组织: 第 1 节, 简要介绍算法(SLVAM)和它的一个子算法(相对熵方法, the cross-entropy method); 第 2 节, 构造算法 ISLVAM, 并证明它的渐近收敛性; 最后在第 3 节, 给出 2 个算例说明算法 ISLVAM 的有效性.

1 连续优化问题的随机水平值逼近算法

对于一般的连续全局优化问题, 我们首先提出了一种箱约束下全局优化的水平值估计算法[13], 进而研究了一般不等式约束下的情况[14], 接着我们又应用 Markov 链的 Monte-Carlo 随机模拟方法, 建立了随机水平值逼近算法(SLVAM), 数值试验表明该算法对于可行域丰满性较好的连续全局优化问题有很高的精度和较好的效率. 为了对本文的算法有一个全面的理解, 本节对算法(SLVAM)和它的一个子算法(相对熵方法, the cross-entropy method)作个简单介绍.

首先将约束全局优化问题

$$\min_{\boldsymbol{x} \in X} f(\boldsymbol{x}) \tag{5}$$

转化为无约束问题, 其中 $X = \{\boldsymbol{x} \in R^n: g(\boldsymbol{x}) \leqslant 0\}$, $f: R^n \to \mathbf{R}$, $g: R^n \to R^m$ 连续.

我们利用非连续精确罚问题

$$\min_{\boldsymbol{x} \in R^n} P(\boldsymbol{x}), \tag{6}$$

其中 $P(\boldsymbol{x}) = f(\boldsymbol{x}) + \lambda\varphi(\boldsymbol{x})$, $\lambda > 0$ 充分大,

$$p(\boldsymbol{x}) = \begin{cases} 0, & \boldsymbol{x} \in X \\ \delta + d(\boldsymbol{x}), & \delta > 0, \boldsymbol{x} \notin X, \end{cases}$$

$$d(\boldsymbol{x}) = \sum_{j=1}^{m} \max(0, g_j(\boldsymbol{x}))$$

来实现从有约束问题向无约束问题的转化.

令 $F(\boldsymbol{x}) = c + I_{\{P(\boldsymbol{x}) \leqslant c\}}(\boldsymbol{x})(c - P(\boldsymbol{x}))$, 则当 c 大于问题(6)的全局最小值时, 问题(6)等价于

$$\min_{\boldsymbol{x} \in R^n} F(\boldsymbol{x}). \tag{7}$$

针对问题(7)来建立算法 SLVAM, 其主要步骤如下:

算法 SLVAM

S0 初始化. 给定 $\boldsymbol{x}_0 \in X$, $c_0 = f(\boldsymbol{x}_0)$, $\boldsymbol{u}_0$, σ_0, $0.5 \leqslant \alpha \leqslant 0.9$, $0.8 \leqslant \beta \leqslant 0.99$, $\beta_0 = \beta$, $\varepsilon > 0$ 任意小, $T > 1$, $k := 0$.

S1 在 R^n 上依概率密度函数 $N(\boldsymbol{x}; \boldsymbol{u}_k, \sigma_k^2)$ 随机取样 T 个点 $\{\boldsymbol{x}_i^k\}_{i=1}^{T}$, 其中 $\boldsymbol{u}_k = (u_{k1}, u_{k2}, \cdots, u_{kn})$, $\sigma_k^2 = (\sigma_{k1}^2, \sigma_{k2}^2, \cdots, \sigma_{kn}^2)$, $\boldsymbol{u}_k$ 和 σ_k^2 中各分量相互独立.

S2 计算 $F(\boldsymbol{x}_i^k, c_k)$ 和

$$c_{k+1} = \frac{1}{T} \sum_{i=1}^{T} F(\boldsymbol{x}_i^k, c_k),$$

$$v_k = \sum_{i=1}^{T} | F(\boldsymbol{x}_i^k, c_k) - c_k | .$$

令 $S_{k+1} = \{\boldsymbol{x} \in X : P(\boldsymbol{x}) \leqslant c_{k+1}\}$, 令离散水平集 $H_{k+1} = \{\boldsymbol{x}_m^k : P(\boldsymbol{x}_m^k) \leqslant c_{k+1}, m \in \{1, 2, \cdots, T\}\}$.

S3 如果 $v_k < \varepsilon$, 转 S7; 否则进入下一步.

S4 从 $\{\boldsymbol{x}_i^k\}_{i=1}^{T}$ 中产生 T^e 精英样本 $\{\hat{\boldsymbol{x}}_j^k\}_{j=1}^{T^e}$, 它们满足

1) $F(\hat{\boldsymbol{x}}_1^k, c_k) = \min\{F(\boldsymbol{x}_i^k, c_k) : \forall \boldsymbol{x}_i^k \in \{\boldsymbol{x}_i^k\}_{i=1}^{T}\}$,

$F(\hat{\boldsymbol{x}}_2^k, c_k) = \min\{F(\boldsymbol{x}_i^k, c_k) : \forall \boldsymbol{x}_i^k \in \{\boldsymbol{x}_i^k\}_{i=1}^{T} / \{\hat{\boldsymbol{x}}_1^k\}\}$, (8)

$F(\hat{\boldsymbol{x}}_3^k, c_k) = \min\{F(\boldsymbol{x}_i^k, c_k) : \forall \boldsymbol{x}_i^k \in \{\boldsymbol{x}_i^k\}_{i=1}^{T} / \{\hat{\boldsymbol{x}}_1^k, \hat{\boldsymbol{x}}_2^k\}\}$, 依此类推;

2) $P(\hat{\boldsymbol{x}}_{T^e}^k) \leqslant c_{k+1}$, 即 $\hat{\boldsymbol{x}}_{T^e}^k \in S_{k+1}$; (9)

3) $T^e = \max\{i : P(\hat{\boldsymbol{x}}_i) \leqslant c_{k+1}\}$. (10)

S5 更新参数. 令

$$u_{k+1,l} := \frac{1}{T^e} \sum_{j=1}^{T^e} \hat{x}_{j,l}^k, \qquad l = 1, 2, \cdots, n, \tag{11}$$

$$\sigma_{k+1,l}^2 := \frac{1}{T^e} \sum_{j=1}^{T^e} (\hat{x}_{j,l}^k - u_{k+1,l})^2, \qquad l = 1, 2, \cdots, n, \tag{12}$$

并令磨光算子

$$\boldsymbol{u}_{k+1} = \alpha \boldsymbol{u}_{k+1} + (1 - \alpha) \boldsymbol{u}_k, \quad \sigma_{k+1} = \beta_{k+1} \sigma_{k+1} + (1 - \beta_{k+1}) \sigma_k, \tag{13}$$

其中

$$\beta_{k+1} = \beta - \beta \left(1 - \frac{1}{k+1}\right)^q, \qquad 5 \leqslant q \leqslant 10, \ q \text{ 为整数}. \tag{14}$$

S6　令 $k:=k+1$, 返回 S1.

S7　令 $c^*=c_{k+1}$ 为近似全局最小值, $S^*=H_{k+1}$ 为近似全局最小解集. 算法终止.

算法 SLVAM 中 $\mathcal{N}(\boldsymbol{x};\boldsymbol{u}_k,\sigma_k^2)$ 表示 n 维坐标正态分布, 具有如下形式:

$$\mathcal{N}(\boldsymbol{x};\boldsymbol{u},\sigma^2)=\frac{1}{(\sqrt{2\pi})^n\prod_{l=1}^n\sigma_l}\exp\left(-\sum_{l=1}^n\frac{(x_l-u_l)^2}{2\sigma_l^2}\right),$$

其中 $\boldsymbol{x}=(x_1,x_2,\cdots,x_n)^{\mathrm{T}}$, $\boldsymbol{u}=(u_1,u_2,\cdots,u_n)^{\mathrm{T}}$, $\sigma=(\sigma_1,\sigma_2,\cdots,\sigma_n)^{\mathrm{T}}$.

在算法SLVAM 中我们利用了相对熵方法更新取样密度(S4 和 S5), 相对熵方法(the cross-entropy method) 由 R. Y. Rubinstein 提出并应用于连续全局优化问题[15]. 它的主要思想是利用前一次投点的函数值信息构造本次投点的概率密度函数, 使得更多的点落在"重点区域" 内. 文献[15] 利用坐标正态密度函数族 $\mathcal{N}(\boldsymbol{x};\boldsymbol{u}_k,\sigma_k^2)$ 作为取样密度, 每次投点前更新参向量 $\boldsymbol{u}_k$、σ_k, 使得密度函数 $\mathcal{N}(\boldsymbol{x};\boldsymbol{u}_k,\sigma_k^2)$ 在某种意义上与目标函数 $f(\boldsymbol{x})$ 更相似. 这种思想来源于极小化相对熵

$$d(f,g)=\int_\Omega g(\boldsymbol{x})\ln f(\boldsymbol{x})\mathrm{d}\boldsymbol{x},$$

因而被称为相对熵方法.

2　凸二次整数极小化的随机水平值逼近算法

为行文方便, 记 $\mu(A)$ 为集合 A 的元素的个数, 分别记 $P(\boldsymbol{x})$、$Q(\boldsymbol{u})$ 为概率密度 $p(\boldsymbol{x})$、$q(\boldsymbol{u})$ 所对应的分布函数.

我们将算法 SLVAM 的基本思想应用到求解凸二次整数极小化问题(1), 构造算法 ISLVAM 如下.

算法 ISLVAM

s0　初始化. 给定初始分布 $P^0(\boldsymbol{x})=P(\boldsymbol{x};\boldsymbol{a}_0,\boldsymbol{v}_0)$, $Q^0(\boldsymbol{u})=Q(\boldsymbol{u};\boldsymbol{b}_0,\boldsymbol{w}_0)$, α, $\beta_0=\beta$, $(0.5<\alpha<0.8, 0.8<\beta<0.99)$, $\varepsilon>0$ 和 $c_0=+\infty$, $d_0=-\infty$, $\Lambda_0=M$ 充分大, $k:=0$.

s1　依分布列 $P^k(\boldsymbol{x})$, $\boldsymbol{x}\in Z^n$ 取 T 个样本点记为 $D^k=\left\{\boldsymbol{X}_i^k\right\}_{i=1}^{\mathrm{T}}$. 记 $S_{\mathrm{pf}}^k=\left\{\boldsymbol{x}\in S_{\mathrm{pf}}:f_{\mathrm{p}}(\boldsymbol{x})\leqslant c_k\right\}$. 计算 $f_{\mathrm{p}}(\boldsymbol{X}_i^k)$, $i=1,2,\cdots,T$, 并令 $H_{\mathrm{pf}}^k=\left\{\boldsymbol{X}_i^k\in D^k\cap S_{\mathrm{pf}}^k:f_{\mathrm{p}}(\boldsymbol{X}_i^k)\leqslant c_k,\ i=1,2,\cdots,T\right\}$.

s2　计算

$$c_{k+1}=\frac{1}{\mu(H_{\mathrm{pf}}^k)}\sum_{j=1}^{\mu(H_{\mathrm{pf}}^k)}f_{\mathrm{p}}(\boldsymbol{X}_j^k),\qquad \boldsymbol{X}_j^k\in H_{\mathrm{pf}}^k \tag{15}$$

及

$$\delta_{k+1}=\sum_{j=1}^{\mu(H_{\mathrm{pf}}^k)}|f_{\mathrm{p}}(\boldsymbol{X}_j^k)-c_{k+1}|,\qquad \boldsymbol{X}_j^k\in H_{\mathrm{pf}}^k, \tag{16}$$

如果 $\delta_{k+1}=0$, 转 s8, 否则进入下一步.

s3　(求对偶问题 DF) 依概率分布函数 $Q^k(\boldsymbol{u})$ 在 R_+^m 中随机取 T 个样本点, 记为 $E^k=\left\{\boldsymbol{U}_i^k\right\}_{i=1}^T$. 计算 $f_{\mathrm{d}}(\boldsymbol{U}_i^k)$, $i=1,2,\cdots,T$, 令 $S_{\mathrm{df}}^k=\left\{\boldsymbol{u}\in S_{\mathrm{df}}:f_{\mathrm{d}}(\boldsymbol{u})\geqslant d_k\right\}$ 及 $H_{\mathrm{df}}^k=\left\{\boldsymbol{U}_j^k\in S_{\mathrm{df}}^k\cap E^k:f_{\mathrm{d}}(\boldsymbol{U}_j^k)\geqslant d_k\right\}$.

s4　如果 $\Lambda_k<\varepsilon$, 令 $d_{k+1}=d_k$; 否则, 计算

$$d_{k+1}=\frac{1}{\mu(H_{\mathrm{df}}^k)}\sum_{j=1}^{\mu(H_{\mathrm{df}}^k)}f_{\mathrm{d}}(\boldsymbol{U}_j^k),\qquad \boldsymbol{U}_j^k\in H_{\mathrm{df}}^k \tag{17}$$

及

$$\Lambda_{k+1}=\sum_{j=1}^{\mu(H_{\mathrm{df}}^k)}|f_{\mathrm{d}}(\boldsymbol{U}_j^k)-d_{k+1}|,\qquad \boldsymbol{U}_j^k\in H_{\mathrm{df}}^k. \tag{18}$$

s5 如果 $|c_{k+1}-d_{k+1}|<\varepsilon$, 转向 s8; 否则转 s6.

s6 更新取样分布.

s6.1 更新分布

$$P_{k+1}(\boldsymbol{x})=P(\boldsymbol{x};\boldsymbol{a}(\boldsymbol{X}_j^k),\boldsymbol{v}(\boldsymbol{X}_j^k)),\qquad \boldsymbol{X}_j^k\in H_{\mathrm{pf}}^k.$$

s6.2 更新分布

$$Q_{k+1}(\boldsymbol{u})=Q(\boldsymbol{u};\boldsymbol{b}(\boldsymbol{U}_j^k),\boldsymbol{w}(\boldsymbol{U}_j^k)),\qquad \boldsymbol{U}_j^k\in H_{\mathrm{df}}^k.$$

s7 如果另外的终止条件不满足, 令 $k:=k+1$, 转 s1, 否则进入下一步.

s8 令 $\boldsymbol{x}^*=\boldsymbol{X}^*$ $(\boldsymbol{X}^*\in H_{\mathrm{pf}}^k)$ 为近似最优解, $c^*=f_{\mathrm{p}}(\boldsymbol{X}^*)$ 为近似最优值. 算法终止.

实现算法中我们选择 n 维格点 Cauchy 密度作为取样密度函数族. 即令 $p(\boldsymbol{x})$ 与 $q(\boldsymbol{u})$ 具有如下形式:

$$\rho(\boldsymbol{y};\mu,\boldsymbol{h})=\prod_{l=1}^{n}\frac{1}{\pi h_l(1+((y_l-\mu_l)/h_l)^2)}. \tag{19}$$

相应于密度 $\rho(\boldsymbol{y})$ 的边际分布函数是

$$\rho_l(t)=\frac{1}{\pi}\arctan\frac{t-u_l}{h_l}+0.5,\qquad l=1,2,\cdots,n.$$

这样 s1 和 s6 分别可以用如下 2 个子程序来实现:

子程序 1 参见文献[16] (实现 s1)

e1 令 $i=1$, $l=1$.

e2 生成 $\xi\in U[0,1]$, 取 $X_{i,l}^k=[h_l]([\tan(\xi-0.5)\pi]+1)+[\mu_l]$, 其中算子 $[\cdot]$ 表示取整, $U[0,1]$ 表示 $[0,1]$ 上的均匀分布.

e3 若 $l<n$, $l:=l+1$ 转 e2.

e4 如果 $i=T$ 终止; 否则 $i:=i+1$ 转 e2.

子程序 2 (the cross-entropy method) (实现 s6)

s6.1

$$a_{k+1,l}=\frac{1}{\mu(H_{\mathrm{pf}}^k)}\sum_{j=1}^{\mu(H_{\mathrm{pf}}^k)}X_{j,l}^k,\qquad l=1,2,\cdots,n;\ \boldsymbol{X}_j^k\in H_{\mathrm{pf}}^k, \tag{20}$$

$$v_{k+1,l}=\gamma\cdot\max_{i,j}\left\{|X_{i,l}^k-X_{j,l}^k|\right\},$$

$$l=1,2,\cdots,n;\ \boldsymbol{X}_i^k,\ \boldsymbol{X}_j^k\in H_{\mathrm{pf}}^k;\ 0<\gamma<1. \tag{21}$$

并磨光

$$\boldsymbol{a}_{k+1}=\alpha\boldsymbol{a}_{k+1}+(1-\alpha)\boldsymbol{a}_k,$$

$$\boldsymbol{v}_{k+1}=\beta_{k+1}\boldsymbol{v}_{k+1}+(1-\beta_{k+1})\boldsymbol{v}_k,$$

其中 $$\beta_{k+1}=\beta-\beta\left(1-\frac{1}{k+1}\right)^q,\ 5<q<10,\ q\in\mathbf{Z}.$$

s6.2

$$b_{k+1,l} = \frac{1}{\mu(H_{\mathrm{df}}^k)} \sum_{j=1}^{\mu(L^k)} U_{j,l}^k, \qquad l = 1, 2, \cdots, n;\ \boldsymbol{U}_j^k \in H_{\mathrm{df}}^k. \tag{22}$$

$$w_{k+1,l} = \eta \cdot \max_{i,j} \left\{ | U_{i,l}^k - U_{j,l}^k | \right\},$$

$$l = 1, 2, \cdots, n;\ \boldsymbol{U}_i^k, \boldsymbol{U}_j^k \in H_{\mathrm{df}}^k;\ 0 < \eta < 1. \tag{23}$$

并磨光

$$\boldsymbol{b}_{k+1} = \alpha \boldsymbol{b}_{k+1} + (1 - \alpha) \boldsymbol{b}_k,$$

$$\boldsymbol{w}_{k+1} = \beta_{k+1} \boldsymbol{w}_{k+1} + (1 - \beta_{k+1}) \boldsymbol{w}_k.$$

注　s7 中"另外的终止条件"可以是

$$\theta_k = \max_l \left\{ v_{k,l} \right\} < \varepsilon.$$

下面我们来证明算法 ISLVAM 是渐近收敛的.

引理 2.1　设水平集 $S_{\mathrm{pf}}^k = \left\{ \boldsymbol{x} \in S_{\mathrm{pf}}: f_{\mathrm{p}}(\boldsymbol{x}) \leqslant c_k \right\}$, 则

1) $H_{\mathrm{pf}}^k \subseteq S_{\mathrm{pf}}^k$;

2) 若 $c_{k+1} \leqslant c_k$, 则 $S_{\mathrm{pf}}^{k+1} \subseteq S_{\mathrm{pf}}^k$;

3) 设 $\mu(S_{\mathrm{pf}})$ 有限, 若 S_{pf}^k 非空, 则依概率有

$$H_{\mathrm{pf}}^k \neq \phi.$$

证明　1) 与 2) 显然成立, 下证 3). 在 S_{pf} 中按均匀分布随机取一点 $\boldsymbol{X}$, 该点落在 S_{pf}^k 中的概率为

$$p = P_{\mathrm{r}} \left\{ X \in S_{\mathrm{pf}}^k \right\} = \frac{\mu(S_{\mathrm{pf}}^k)}{\mu(S_{\mathrm{pf}})} < 1.$$

由函数 $f_{\mathrm{p}}(\boldsymbol{x})$ 的凸性和多胞形 $X = \left\{ \boldsymbol{A}\boldsymbol{x} \leqslant \boldsymbol{b} \right\}$ 的凸性知, 集 H_{pf}^{k-1} 的中心 $\boldsymbol{a}_k \in \left\{ f_{\mathrm{p}}(\boldsymbol{x}) \leqslant c_{k-1} \right\} \cap X$. 又由于

$$v_{k,l} = \forall \max_{i,j} | X_{i,l}^{k-1} - X_{j,l}^{k-1} |, \qquad \boldsymbol{X}_i^{k-1}, \boldsymbol{X}_j^{k-1} \in H_{\mathrm{pf}}^{k-1},$$

故由重点取样的性质可知, 按概率密度 $\rho(\boldsymbol{x}; \boldsymbol{a}_k, \boldsymbol{v}_k)$ 在 S_{pf} 中随机取一个点, 该点落在 S_{pf}^k 中的概率为 $p_1 > p$, 同时显然 $p_1 < 1$. 从而直接计算可知, 在 S_{pf} 中按密度为 $\rho(\boldsymbol{x}; \boldsymbol{a}_k, \boldsymbol{v}_k)$ 的分布随机取 T 个点, 至少有一点落在 S_{pf}^k 中, 即 $H_{\mathrm{pf}}^k \neq \phi$ 的概率为

$$P_{\mathrm{r}} \left\{ H_{\mathrm{pf}}^k \neq \phi \right\} = 1 - [1 - p_1]^T > 1 - [1 - p]^T,$$

所以

$$1 \geqslant \lim_{T \to \infty} P_{\mathrm{r}} \left\{ H_{\mathrm{pf}}^k \neq \phi \right\} = \lim_{T \to \infty} [1 - (1 - p_1)^T] > \lim_{T \to \infty} [1 - (1 - p)^T] = 1,$$

即

$$\lim_{T \to \infty} P_{\mathrm{r}} \left\{ H_{\mathrm{pf}}^k \neq \phi \right\} = 1.$$

定理 2.2　由算法 ISLVAM 产生的序列 $\left\{ c_k \right\}$ $(k = 0, 1, 2, \cdots)$ 是单调下降且下有界的.

证明　由 $\boldsymbol{X}_j^k \in H_{\mathrm{pf}}^k \subseteq S_{\mathrm{pf}}^k = \left\{ \boldsymbol{x} \in S_{\mathrm{pf}}: f_{\mathrm{p}}(\boldsymbol{x}) \leqslant c_k \right\}$ 知,

$$c_{k+1} = \frac{1}{\mu(H_{\mathrm{pf}}^k)} \sum_{j=1}^{\mu(H_{\mathrm{pf}}^k)} f_{\mathrm{p}}(\boldsymbol{X}_j^k) \leqslant \frac{1}{\mu(H_{\mathrm{pf}}^k)} \sum_{j=1}^{\mu(H_{\mathrm{pf}}^k)} c_k = c_k,$$

所以序列 $\left\{ c_k \right\}$ 单调下降. 若设 c 是问题 SF 的最优值, 由定理 1.1 和 1.2, 可知 c 也是对偶问题 DF 的最优值, 且 $c_k \geqslant c$, $\forall k = 0, 1, 2, \cdots$. 事实上, 当 $c_k = c$ 时, 如果 $d_k = c$, 则 $| c_k - d_k | = c_k - d_k = 0 < \varepsilon$, 由 S5 算法终止; 而若 $d_k < c$, SF 的水平集为 $H_c^{\mathrm{sf}} = \left\{ \boldsymbol{x} \in S_{\mathrm{sf}}: f_{\mathrm{s}}(\boldsymbol{x}) = c \right\}$, 所以对

于任意 $\boldsymbol{X}_j^k \in \left\{ \boldsymbol{x} \in S_{\mathrm{pf}}: f_{\mathrm{p}}(\boldsymbol{x}) \leqslant c \right\} \subseteq H_c^{\mathrm{sf}}$, 都有 $| f_{\mathrm{p}}(\boldsymbol{X}_j^k) - c | = 0 < \varepsilon$, 由 S2 算法终止. 因此 c 是 $\{ c_k, k = 0, 1, 2, \cdots \}$ 的下界.

引理 2.3 若 $\boldsymbol{x}^*$ 是问题 PF 的解, 则 $\boldsymbol{x}^* \in S_{\mathrm{pf}}^k, k = 0, 1, 2, \cdots$.

证明 由 $S_{\mathrm{pf}}^k = \left\{ \boldsymbol{x} \in S_{\mathrm{pf}}: f_{\mathrm{p}}(\boldsymbol{x}) \leqslant c_k \right\}$ 及 $\boldsymbol{x}^*$ 为问题 PF 的解, 结合引理 2.1 和定理 2.2, 该命题显然成立.

定理 2.4 设 $\boldsymbol{x}^*$ 是 PF 的任一个最优点, $c^* = f_{\mathrm{p}}(\boldsymbol{x}^*)$. 则由算法 ISLVAM 产生的序列 $\{ c_k \}$ 满足

$$P_{\mathrm{r}}\left(\lim_k c_k = c^* \right) = 1. \tag{24}$$

证明 由定理 2.2 知, 序列 $\{ c_k \}$ 单调下降有下界, 因此收敛.

又由引理 2.3 知, $\boldsymbol{x}^* \in S_{\mathrm{pf}}^k, k = 0, 1, 2, \cdots$, 再由引理 2.1 知, $S_{\mathrm{pf}}^k \supseteq S_{\mathrm{pf}}^{k+1}$, 所以极限集

$$S_{\mathrm{pf}}^* = \lim_k S_{\mathrm{pf}}^k$$

存在且非空. 由算法 ISLVAM, 假设序列 $\{ c_k \}$ 收敛到 $\hat{c}$, 则当 $\hat{c} > c^*$ 时, 至少存在一点 $\boldsymbol{x}^* \in S_{\mathrm{pf}}^k$, 由引理 2.1 的 3) 知, $P_{\mathrm{r}}\{ H_{\mathrm{pf}}^k \neq \mathrm{f} \} = 1$, 其中 $H_{\mathrm{pf}}^k = \left\{ \boldsymbol{X}_j^k \in S_{\mathrm{pf}}^k: f_{\mathrm{p}}(\boldsymbol{X}_j^k) \leqslant c_k = \hat{c} \right\}$, 由算法终止条件, 此时有 $c_{k+1} = c^* \leqslant \hat{c}$, 得到矛盾; 若 $\hat{c} < c^*$, 由迭代公式 (15) 及 c^* 是问题 PF 的最小值, 存在 k 使得

$$\begin{aligned} c_{k+1} = \hat{c} &= \frac{1}{\mu(H_{\mathrm{pf}}^k)} \sum_{j=1}^{\mu(H_{\mathrm{pf}}^k)} f_{\mathrm{p}}(\boldsymbol{X}_j^k), \qquad \boldsymbol{X}_j^k \in H_{\mathrm{pf}}^k, \\ &\geqslant \frac{1}{\mu(H_{\mathrm{pf}}^k)} \sum_{j=1}^{\mu(H_{\mathrm{pf}}^k)} c^* = c^*, \end{aligned}$$

也得到矛盾; 所以 $\hat{c} = c^*$.

综上, 我们得到

$$P_{\mathrm{r}}\left(\lim_k c_k = c^* \right) = 1.$$

3 数值实验与结论

本节我们用如下 2 个测试函数来检验算法 ISLVAM 的有效性, 并把测试结果与相应文献所提出的算法的计算结果作对比. 我们的实验环境是: Pentium(R) 4 CPU 2.00 GHz, 256 Mb 内存; Matlab 6.5.

算例 3.1

$$\begin{aligned} &\min f(\boldsymbol{x}) = (x_1 - 3.4)^2 + (x_2 - 3.2)^2 + (x_3 - 4)^2 + (x_4 - 2)^2 + (x_5 - 2)^2 \\ &\text{s.t. } 1 \leqslant x_i \leqslant 9, \ i = 1, 2, 3, 4, 5; \ x_i \in \mathbf{Z}. \end{aligned}$$

应用算法 ISLVAM, 对算例 3.1 进行多次测试. 在测试中, 每轮投点 10 个, 迭代不超过 9 轮, 耗时不超过 0.20 s, 每次都能得到最优解 (3, 3, 4, 2, 2). 而应用文献[17]所提出的算法需要计算函数值 4 729 次, 得到 12 个接近最小值的点, 其中包括最优点.

算例 3.2

$$\begin{aligned} &\min f(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x} - 1)^{\mathrm{T}} \boldsymbol{Q}(\boldsymbol{x} - 1) \\ &\text{s.t. } | x_i | \leqslant L, \ \boldsymbol{x} \in Z^n, \ i = 1, 2, \cdots, n, \end{aligned}$$

其中 $\boldsymbol{Q} \in R^{n \times n}$ 是随机产生的正定矩阵.

文献[18]用改进的混合蚁群算法(HAC)求解,并与 Multi-start 的 HAC 算法及原来的HAC做了比较,分别计算了 $n = 5, 10$ 的情况(计算结果可参见文献[18]表 1). 我们应用算法 ISLVAM,分别计算了 $n = 5, 10, 20, 50, 100, 200$ 的情况,结果依概率收敛到最优点$(1, 1, \cdots, 1)$. 具体计算参数如表 1. 起始点选为$(3, 3, \cdots, 3)^{\mathrm{T}}$.

表 1　　计算结果

n	L	T	迭代次数 N
5	100	100	10
10	100	100	18
20	100	500	26
50	100	1 000	48
100	100	2 000	73
200	100	2 000	109

将本例稍作一般化

$$\min f(\boldsymbol{x}) = \frac{1}{2}(\boldsymbol{x} - \boldsymbol{e})^{\mathrm{T}}\boldsymbol{Q}(\boldsymbol{x} - \boldsymbol{e})$$

$$\text{s.t. } |x_i| \leqslant L,\ L > \|\boldsymbol{e}\|,\ \boldsymbol{x} \in Z^n,\ i = 1, 2, \cdots, n,$$

其中 $\boldsymbol{Q} \in R^{n\times n}$ 是随机产生的正定矩阵,$\boldsymbol{e} \in R^n$ 为给定向量. 显然该问题有极小点 $\boldsymbol{x}^* = \Upsilon(\boldsymbol{e})$,算子 $\Upsilon(\boldsymbol{e})$ 表示对 $\boldsymbol{e}$ 的各个分量 e_i 取最接近的整数. 计算表明算法 ISLVAM 对于这类问题具有和算例 3.2 同样的效率.

试验结果表明,算法 ISLVAM 在实践中具有与理论结果相符的渐近收敛性,并有较高的运行效率. 事实上,我们的算法只需稍做改动,即可以应用到更一般的凸二次离散规划问题. 但值得注意的是,对于 $\boldsymbol{x} \in \{0, 1\}^n$ 的情况,可以直接利用二项分布取样,应用 Markov 极限理论证明算法收敛性,参见文献[12].

[参　考　文　献]

[1] Wolsey Laurence A. Integer Programming[M]. New York: John Wiley and Sons, Inc. 1998.

[2] Land A H, Doig A G. An automatic method for solving discrete problems[J]. Econometrica, 1960, **28**(3): 497-520.

[3] Dakin R J. A tree search algorithm for mixed integer programming problems[J]. Computer Journal, 1965, **8**(3): 250-255.

[4] Dantzig G B, Fulkerson D R, Johnson S. Solution of a large scale traveling salesman problem[J]. Operations Research, 1954, **2**(4): 393-410.

[5] Gomory R E. An algorithm for the mixed integer problem[R]. RM-2597, The Rand Corporation, 1960.

[6] Derpich I, Vera J R. Improving the efficiency of the Branch and Bound algorithm for integer programming based on "flatness" information[J]. European Journal of Operational Research, 2006, **174**(1): 92-101.

[7] HUA Zhong-sheng, HUANG Fei-hua. An effective genetic algorithm approach to large scale mixed integer programming problems[J]. Applied Mathematics and Computation, 2006, **174**(2): 897-909.

[8] Bosio Sandro, Righini Giovanni. Computational approaches to a combinatorial optimization problem arising from text classification[J]. Computers and Operations Research, 2007, **34**(7): 1910-1928.

[9] Arostegui Jr Marvin A, Kadipasaoglu Sukran N, Khumawala Basheer M. An empirical comparison of Tabu search, simulated annealing, and genetic algorithms for facilities location problems[J]. International Journal of Production Economics, 2006, **103**(2): 742-754.

[10] Dino Ahr, Gerhard Reinelt. A Tabu search algorithm for the min-max k-Chinese postman problem [J]. Computers and Operations Research, 2006, **33**(12): 3403-3422.

[11] Mokhtar S B, Hanif D S, Shetty C M. Nonlinear Programming, Theory and Algorithms [M]. Hoboken, Canada: John Wiley and Sons, Inc. 1993, 199-233.

[12] De Boer P-T, Kroese D P, Mannor S, et al. A tutorial on the cross-entropy method[J]. Annals of Operations Research, 2005, **134**(1): 19-67.

[13] 邬冬华, 俞武扬, 田蔚文. 一种求约束总极值的水平值估计方法[J]. 应用数学和力学, 2006, **27**(7): 874-882.

[14] 彭拯, 邬冬华, 田蔚文. 约束全局优化的水平值估计算法[J]. 计算数学, 2007, **29**(3): 293-304.

[15] Rubinstein R Y. The cross-entropy method for combinatorial and continuous optimization[J]. Methodology and Computing in Applied Probability, 1999, **1**(2): 127-190.

[16] Sheldon M Ross. Simulation [M]. 3rd Ed. Beijing, China: Posts and Telecom Press, 2006: 45-47.

[17] 万国成, 田翔, 任震. 一类非线性整数规划问题的计算机求解[J]. 计算机工程与应用, 2002, **38**(16): 80-83.

[18] 林锦, 朱文兴. 凸整数规划问题的混合蚁群算法[J]. 福州大学学报(自然科学版), 1999, **27**(6): 5-9.

Stochastic Level-Value Approximation for Quadratic Integer Convex Programming

PENG Zheng[1,2], WU Dong-hua[1]

(1. Department of Mathematics, Shanghai University, Shanghai 200444, P.R. China;
2. Department of Mathematics, Hunan Institute of Technology and Science, Yueyang, Hunan 414006, P.R. China)

Abstract: A stochastic level value approximating method for quadratic integer convex minimizing problem was proposed. This method applies the importance sampling technique, and uses the main idea of the cross-entropy method to update the sample density functions. The asymptotic convergence of this algorithm was also proved, and some numerical results to illuminate its efficiency was reported.

Key words: quadratic integer convex programming; stochastic level value approximation; cross-entropy method; asymptotic convergence